Amendments to the Claims

Please amend the claims as follows:

1. (Currently Amended)   A method for modeling a legacy computer system comprising:

identifying incidents of applications of the legacy computer system that output data;

defining a control flow graph of the output incidents; ~~and~~

wherein the incidents comprise report ~~commands.~~commands; and

wherein the control flow graph comprises plural nodes having associated arcs, at least some of the plural nodes associated with an output incident.

2. (Original) The method of Claim 1 further comprising:

identifying the value or type of the data fields associated with each output incident; and

attaching the value or type to the control flow graph.

3. (Original) The method of Claim 2 wherein identifying the value or type further comprises:

identifying output incidents of invariant data fields; and

attaching the value of each invariant data field to its associated control flow graph incident.

4. (Original) The method of Claim 2 wherein identifying the value or type further comprises:

identifying output incidents of variant data fields; and

attaching the type of each variant data field to its associated control flow graph incident.

5. (Allowed) A method for modeling a legacy computer system comprising:

identifying incidents of applications of the legacy computer system that output data;

defining a control flow graph of the output incidents; and

wherein the control flow graph comprises plural nodes having associated arcs, each node associated with an output incident.

6.   (Allowed)   The method of Claim 5 wherein a complete control flow graph of the application (N,A) is used to compute a directed graph (NR, AR) wherein:

n comprises a node in NR if n, an element of N, starts an output process, stops an output process or outputs data; and

<n1, nm> comprises an arc in AR if n1 and nm are in NR and a sequence of arcs <n1, n2>, <n2, n3>,. . .,<nm-1, nm> is in A such that for i from 2 to m-1, ni is not in NR.

7.   (Allowed)   The method of Claim 6 further comprising:

defining the control flow graph as a formal grammar that describes the flow paths from each start command to the associated stop commands.

8.   (Allowed)   A method for modeling a legacy computer system comprising:

identifying incidents of applications of the legacy computer system that output data;

defining a control flow graph of the output incidents;

associating the incidents with an Extensible Markup Language schema; and

creating a specification to modify the legacy computer system applications to provide output in Extensible Markup Language format.

9.   (Allowed)   The method of Claim 8 further comprising:

automatically modifying the legacy computer system applications in accordance with the specification.

10.   (Currently Amended)   A system for modeling an output application of a legacy computer system comprising:

a modeling engine interfaced with the legacy computer system, the modeling engine operable to analyze an application loaded on the legacy computer system to identify incidents within the application that output data from the legacy computer system;

a control flow graph of the output incidents within the application; ~~and~~

wherein the incidents comprise report ~~commands.~~commands; and

wherein the control flow graph comprises plural nodes having associated arcs, at least some of the plural nodes associated with an output incident.

11.   (Canceled)

12.   (Allowed)   A system for modeling an output application of a legacy computer system comprising:

a modeling engine interfaced with the legacy computer system, the modeling engine operable to analyze an application loaded on the legacy computer system to identify incidents within the application that output data from the legacy computer system;

a control flow graph of the output incidents within the applications;

wherein the control flow graph comprises plural nodes, each node associated with an output incident;

wherein a complete control flow graph of the application (N,A) is used to compute a directed graph (NR, AR) wherein:

n comprises a node in NR if n, an element of N, starts an output process, stops an output process or outputs data; and

<n1, nm> comprises an arc in AR if n1 and nm are in NR and a sequence of arcs <n1, n2>, <n2, n3>,. . .,<nm-1, nm> is in A such that for i from 2 to m-1, ni is not in NR.

13.   (Currently Amended)   ~~The system of claim 10~~A system for modeling an output application of a legacy computer system comprising:

a modeling engine interfaced with the legacy computer system, the modeling engine operable to analyze an application loaded on the legacy computer system to identify incidents within the application that output data from the legacy computer system;

a control flow graph of the output incidents within the application;

wherein the incidents comprise report commands; and

wherein the control flow graph of the output incidents comprises a formal grammar that describes the flow paths from each start command to the associated stop commands.

14.   (Currently Amended)   ~~The system of Claim 10 further comprising~~A system for modeling an output application of a legacy computer system comprising:

a modeling engine interfaced with the legacy computer system, the modeling engine operable to analyze an application loaded on the legacy computer system to identify incidents within the application that output data from the legacy computer system;

a control flow graph of the output incidents within the application;

wherein the incidents comprise report commands; and

a graphical user interface in communication with the modeling engine, the graphical user interface operable to display the control flow graph formal grammar and the incidents.

15.   (Allowed)   The system of Claim 14 wherein the graphical user interface further communicates with a mapping engine and an Extensible Markup Language schema, the mapping engine operable to map the incidents of the applications with the control flow graph formal grammar and the Extensible Markup Language schema.

16.   (Allowed)   A method for modeling a legacy computer system comprising:

defining a control flow graph of output incidents of applications of a legacy computer system;

wherein the control flow graph comprises plural nodes having associated arcs, each node associated with an output incident; and

wherein a complete control flow graph of the application (N,A) is used to compute a directed graph (NR, AR) wherein:

n comprises a node in NR if n, an element of N, starts an output process, stops an output process or outputs data; and

<n1, nm> comprises an arc in AR if n1 and nm are in NR and a sequence of arcs <n1, n2>, <n2, n3>,. . .,<nm-1, nm> is in A such that for i from 2 to m-1, ni is not in NR.

17.   (Currently Amended)   ~~The method of Claim 1,~~A method for modeling a legacy computer system comprising:

identifying incidents of applications of the legacy computer system that output data;

defining a control flow graph of the output incidents;

wherein the incidents comprise report commands; and

wherein identifying incidents of applications of the legacy computer system comprises identifying incidents of applications of the legacy computer system within the source code of the applications.

18.   (Canceled)

19.   (Previously Presented)   The system of Claim 10, wherein the modeling engine is operable to identify the incidents within the source code of the application.

20.   (Canceled)

21.   (Canceled)

22.   (Canceled)

23.   (Canceled)

24.   (Canceled)

25.   (Canceled)

26.   (Canceled)

27.   (Canceled)

28.   (Previously Presented)   The method of Claim 5, further comprising:

identifying the value or type of the data fields associated with each output incident; and

attaching the value or type to the control flow graph.

29.   (Previously Presented)   The method of Claim 28, wherein identifying the value or type further comprises:

identifying output incidents of invariant data fields; and

attaching the value of each invariant data field to its associated control flow graph incident.

30.   (Previously Presented)   The method of Claim 28, wherein identifying the value or type further comprises:

identifying output incidents of variant data fields; and

attaching the type of each variant data field to its associated control flow graph incident.

31.   (Previously Presented)   The method of Claim 5, further comprising:

associating the incidents with an Extensible Markup Language schema; and

creating a specification to modify the legacy computer system applications to provide output in Extensible Markup Language format.

32.   (Previously Presented)   The method of Claim 31, further comprising automatically modifying the legacy computer system applications in accordance with the specification.